lpm_abs Megafunction

User Guide

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
www.altera.com

Software Version: 7.0
Document Version: 2.2
Document Date: March 2007

Copyright © 2007 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

I.S. EN ISO 9001

UG-MF91204-2.2

Contents

About this User Guide v

Document Revision History v
How to Contact Altera v
Typographic Conventions vi

Chapter 1. About this Megafunction

Device Family Support 1–1
Introduction 1–1
Features 1–2
General Description 1–2
Common Applications 1–3
Resource Utilization and Performance 1–3
Software Requirements 1–4

Chapter 2. Getting Started

System Requirements 2–1
Mega Wizard Plug-In Manager Customization 2–1
Using the MegaWizard Plug-In Manager 2–1
Inferring Megafunctions from HDL Code 2–4
Instantiating Megafunctions in HDL Code 2–4
Identifying a Megafunction after Compilation 2–5
Simulation 2–5
Quartus II Simulation 2–5
EDA Simulation 2–6
SignalTap II Embedded Logic Analyzer 2–6
Design Example: 4-Bit Absolute Function 2–6
Design Files 2–6
Example 2–6
Generate a 4-Bit Absolute Function 2–7
Implement the 4-Bit Absolute Value Function 2–11
Functional Results—Simulate the 4-Bit Absolute Value in Quartus 2–11
Functional Results—Simulate the 4-Bit Absolute Value in ModelSim-Altera 2–12
Conclusion 2–15

Chapter 3. Specifications

Ports and Parameters 3–1

About this User Guide

Document Revision History

The table below displays the revision history for the chapters in this User Guide.

Document Date and Version	Changes Made	Summary of Changes
March 2007 v2.2	Added Cyclone III support to the list of supported devices.	—
December 2006 v2.1	Added Stratix III support to the list of supported devices.	—
May 2006 v2.0	Updated for Quartus 6.0 software release.	—
September 200 v1.0	Initial release.	—

How to Contact Altera

For the most up-to-date information about Altera® products, go to the Altera world-wide web site at www.altera.com. For technical support on this product, go to www.altera.com/mysupport. For additional information about Altera products, consult the sources shown below.

Information Type	USA & Canada	All Other Locations
Technical support	www.altera.com/mysupport/	altera.com/mysupport/
	(800) 800-EPLD (3753) (7:00 a.m. to 5:00 p.m. Pacific Time)	(408) 544-7000 *(1)* (7:00 a.m. to 5:00 p.m. Pacific Time)
Product literature	www.altera.com	www.altera.com
Altera literature services	lit_req@altera.com *(1)*	lit_req@altera.com *(1)*
Non-technical customer service	(800) 767-3753 (7:00 a.m. to 5:00 p.m. Pacific Time)	(408) 544-7000 (7:30 a.m. to 5:30 p.m. Pacific Time)
FTP site	ftp.altera.com	ftp.altera.com

Note to table:
(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

This document uses the typographic conventions shown below.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Command names, dialog box titles, checkbox options, and dialog box options are shown in bold, initial capital letters. Example: **Save As** dialog box.
bold type	External timing parameters, directory names, project names, disk drive names, filenames, filename extensions, and software utility names are shown in bold type. Examples: $\mathbf{f_{MAX}}$, **\qdesigns** directory, **d:** drive, **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Document titles are shown in italic type with initial capital letters. Example: *AN 75: High-Speed Board Design.*
Italic type	Internal timing parameters and variables are shown in italic type. Examples: t_{PIA}, $n + 1$. Variable names are enclosed in angle brackets (< >) and shown in italic type. Example: *<file name>*, *<project name>***.pof** file.
Initial Capital Letters	Keyboard keys and menu names are shown with initial capital letters. Examples: Delete key, the Options menu.
"Subheading Title"	References to sections within a document and titles of on-line help topics are shown in quotation marks. Example: "Typographic Conventions."
`Courier type`	Signal and port names are shown in lowercase Courier type. Examples: `data1`, `tdi`, `input.` Active-low signals are denoted by suffix `n`, e.g., `resetn`. Anything that must be typed exactly as it appears is shown in Courier type. For example: `c:\qdesigns\tutorial\chiptrip.gdf`. Also, sections of an actual file, such as a Report File, references to parts of files (e.g., the AHDL keyword `SUBDESIGN`), as well as logic function names (e.g., `TRI`) are shown in Courier.
1., 2., 3., and a., b., c., etc.	Numbered steps are used in a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ● •	Bullets are used in a list of items when the sequence of the items is not important.
✓	The checkmark indicates a procedure that consists of one step only.
	The hand points to information that requires special attention.
	A caution calls attention to a condition or possible situation that can damage or destroy the product or the user's work.
	A warning calls attention to a condition or possible situation that can cause injury to the user.
↵	The angled arrow indicates you should press the Enter key.
	The feet direct you to more information on a particular topic.

Chapter 1. About this Megafunction

Device Family Support

The lpm_abs megafunction supports the following target Altera® device families:

- Stratix® III
- Stratix II
- Stratix II GX
- Stratix
- Stratix GX
- Cyclone® III
- Cyclone II
- HardCopy II
- MAX® II
- ACEX® 1K
- APEX™ II
- APEX 20KC
- APEX 20KE
- APEX 20K
- FLEX 10K®
- FLEX 10KA
- FLEX 10KE
- FLEX 6000
- MAX 3000A
- MAX 7000AE
- MAX 7000B
- MAX 7000S

Introduction

As design complexities increase, use of vendor-specific intellectual property (IP) blocks has become a common design methodology. Altera provides parameterizable megafunctions that are optimized for Altera device architectures. Using megafunctions instead of coding your own logic saves valuable design time. Additionally, the Altera-provided functions may offer more efficient logic synthesis and device implementation. You can scale the megafunction's size by simply setting parameters.

Features

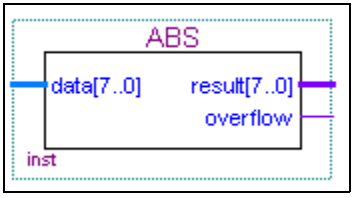
The lpm_absmegafunction implements an absolute value function and offers additional features, which include:

- Parameterizable input data widths
- Support for overflow output

General Description

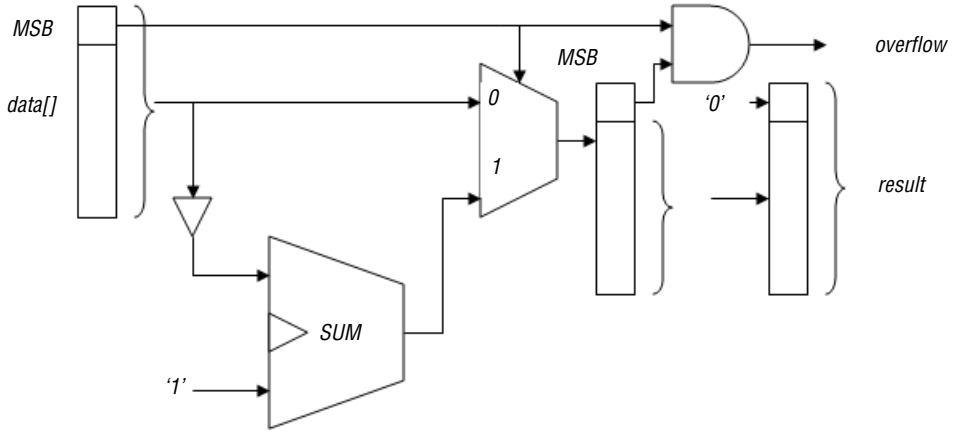
The lpm_absmegafunction is one of the arithmetic megafunctions provided by the Quartus® II software MegaWizard® Plug-In Manager. The lpm_absmegafunction lets you implement an absolute value function that calculates the absolute value of an input. The overflow output is used to indicate that the positive equivalent of an input value does not exist.

Figure 1–1. lpm_abs Megafunction Symbol



A two's complement number system is used to represent the input. In the two's complement system, there is one more negative number than there are positive numbers. Because of this asymmetric range, the negative number $-2^{(\text{the bit width of the data[] -1})}$ does not have a positive equivalent. For a 4-bit width, the possible range of values are from -8 or $(-2^{(4-1)})$ to $+7$ or $(2^{(4-1)} -1)$. The lpm_absmegafunction generates an overflow indication for an input value of –8. All other negative numbers have equivalent positive representations.

Figure 1–2. lpm_abs Megafunction Circuit



In Figure 1–2, the topmost or most significant bit (MSB) of the input (`data[]`) is the sign bit. For negative numbers, the lpm_absmegafunction calculates the absolute value of the input by finding the bitwise complement of the input and adding 1 to this complement.

Common Applications

The absolute value function is a basic operation in computer graphics and scientific calculation applications such as math coprocessors, DSP algorithms, embedded arithmetic co-processors, data processing, and data control. Many possible algorithms and implementations of an absolute value function can be used in an FPGA. The lpm_absmegafunction is an efficient way to implement an absolute value function in Altera FPGAs and is optimized for implementation in all Altera devices.

Resource Utilization and Performance

The lpm_absmegafunction is implemented using logic elements (LEs) or adaptive logic modules (ALMs) in Altera devices.

Table 1–1 summarizes the resource use of an lpm_absfunction when implementing an absolute value function with the default settings in the MegaWizard Plug-In Manager.

Table 1–1. Resource Usage for the lpm_abs Megafunction

Device Family	Width	Logic Use
Stratix II	16	17 ALUTS
Stratix, Stratix GX	16	17 logic elements
Cyclone II, Cyclone	16	17 logic elements

Software Requirements

The lpm_abs megafunction is provided with the Quartus II software. You must install the Quartus II software to use this megafunction.

Chapter 2. Getting Started

System Requirements

The instructions in this section require the following hardware and software:

- A PC running either Windows NT/2000/XP, Red Hat Linux 7.3 or 8.0, Red Hat Linux Enterprise 3, *or* an HP workstation running the HP-UX version 11.0 operating system, *or* a Sun workstation running the Solaris 8 or 9 operating system
- Quartus® II software version 4.1 or later

Mega Wizard Plug-In Manager Customization

The MegaWizard® Plug-In Manager creates or modifies design files that contain custom megafunction variations that can then be instantiated in a design file. The MegaWizard Plug-In Manager provides a wizard that lets you specify options for the lpm_abs megafunction. You can use the MegaWizard Plug-In Manager to set the lpm_abs megafunction features for each absolute value function in the design.

Start the MegaWizard Plug-In Manager in one of the following ways:

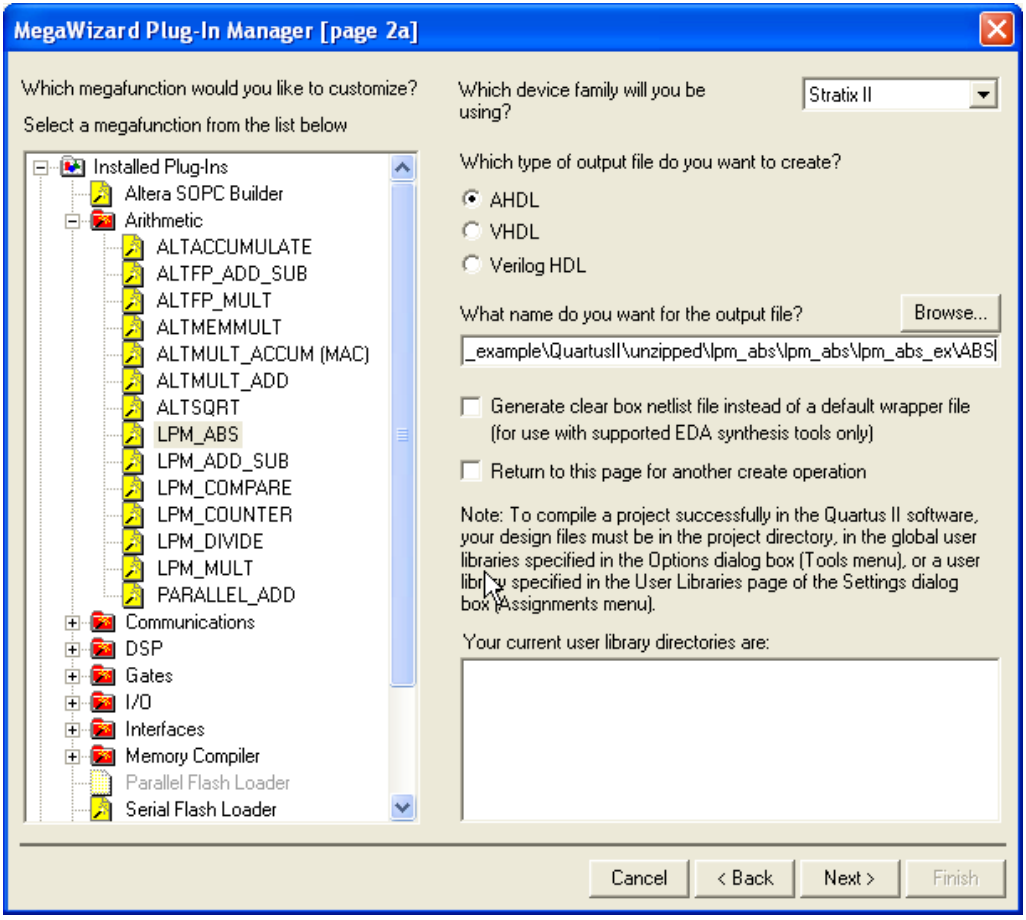
- On the Tools menu, choose the **MegaWizard Plug-In Manager** command.
- When working in the Block Editor, click **MegaWizard Plug-In Manager** in the Symbol window.
- Start the stand-alone version of the **MegaWizard Plug-In Manager** by typing the following command at the command prompt:
  `qmegawiz↵`

Using the MegaWizard Plug-In Manager

This section details descriptions of the options available on the individual pages of the lpm_abs megafunction wizard.

On Page 2a, you select the lpm_abs megafunction from the Arithmetic category, select the device you intend to use, the type of output file you want to create (Verilog, VHDl, or AHDL), and what you want to name the output file. You also have the option to enable the generation of a clear-box netlist for this megafunction (Figure 2–1).

Figure 2–1. MegaWizard Plug-In Manager [Page 2a]


MegaWizard Plug-In Manager [page 2a]
Which megafunction would you like to customize?
Select a megafunction from the list below
Installed Plug-Ins
Altera SOPC Builder
Arithmetic
ALTACCUMULATE
ALTFP_ADD_SUB
ALTFP_MULT
ALTMEMMULT
ALTMULT_ACCUM (MAC)
ALTMULT_ADD
ALTSQRT
LPM_ABS
LPM_ADD_SUB
LPM_COMPARE
LPM_COUNTER
LPM_DIVIDE
LPM_MULT
PARALLEL_ADD
Communications
DSP
Gates
I/O
Interfaces
Memory Compiler
Parallel Flash Loader
Serial Flash Loader
Which device family will you be using?
Stratix II
Which type of output file do you want to create?
AHDL
VHDL
Verilog HDL
What name do you want for the output file?
Browse...
_example\QuartusII\unzipped\lpm_abs\lpm_abs\lpm_abs_ex\ABS
Generate clear box netlist file instead of a default wrapper file (for use with supported EDA synthesis tools only)
Return to this page for another create operation
Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).
Your current user library directories are:
Cancel
< Back
Next >
Finish

On Page 3 of the lpm_abs megafunction wizard, you enable the overflow signal option and specify the bit width of the inputs/outputs of this megafunction. Figure 2–2 shows Page 3 of the lpm_abs wizard.

Figure 2–2. MegaWizard Plug-In Manager - LPM_ABS [Page 3 of 5]

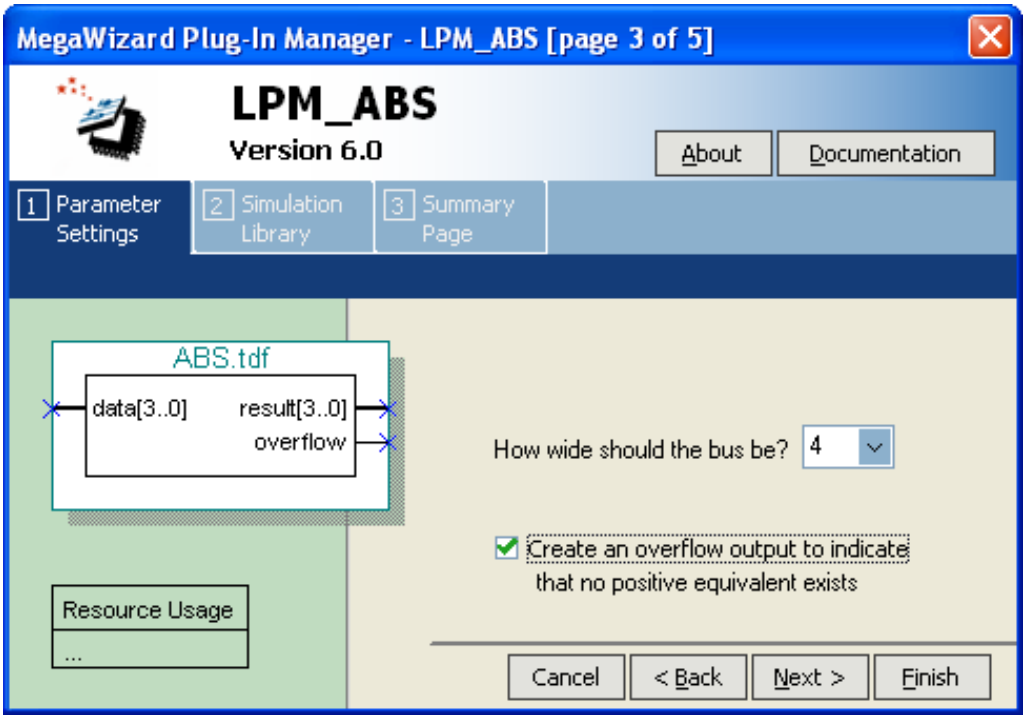


Table 2–1 shows the features and settings of the lpm_abs megafunction. Use this table with hardware descriptions of the features to determine the appropriate settings.

Table 2–1. lpm_abs MegaWizard Plug-in Manager Page 1 Options

Function	Description
How wide should the bus be?	Specify the input and output bus widths value of the function. See the `WIDTH` parameter description in Table 3–3.
Create an overflow output?	Specify the overflow output to indicate that no positive equivalent exists. See the `overflow` port description in Table 3–2.

Inferring Megafunctions from HDL Code

Synthesis tools, including the Quartus II integrated synthesis, recognize certain types of HDL code and automatically infer the appropriate megafunction when a megafunction will provide optimal results. The Quartus II software uses the Altera® megafunction code when compiling your design, even though you did not specifically instantiate the megafunction. The Quartus II software infers megafunctions because they are optimized for Altera devices, so the area and/or performance may be better than generic HDL code. Additionally, you must use megafunctions to access certain Altera architecture-specific features—such as memory, DSP blocks, and shift registers—that generally provide improved performance compared with basic logic elements.

Refer to the appropriate chapter in volume 1 of the *Quartus II Handbook* for specific information about your particular megafunction.

Instantiating Megafunctions in HDL Code

When you use the MegaWizard Plug-In Manager to set up and parameterize a megafunction, it creates either a VHDL or Verilog HDL wrapper file that instantiates the megafunction (a black-box methodology). For some megafunctions, you can generate a fully synthesizable netlist for improved results with EDA synthesis tools, such as Synplify and Precision RTL Synthesis (a clear-box methodology). Both clear- and black-box methodologies are described in the volume 1 of the *Quartus II Handbook*.

Identifying a Megafunction after Compilation

The Quartus II software performs analysis and elaboration to build the structure of your design during compilation. To locate your megafunction in the Project Navigator window, expand the compilation hierarchy and find the megafunction by its name.

To search for node names within the megafunction (using the Node Finder), in the **Look in** box click **Browse (…)** and select the megafunction in the **Hierarchy** box.

Simulation

The Quartus II Simulation tool provides an easy-to-use, integrated solution for performing simulations. The following sections describe the simulation options.

Quartus II Simulation

With the Quartus II Simulator, you can perform two types of simulations: functional and timing. A functional simulation in the Quartus II program enables you to verify the logical operation of your design without taking into consideration the timing delays in the FPGA. This simulation is performed using only RTL code. When performing a functional simulation, add only signals that exist before synthesis. You can find these signals with the Registers: pre-synthesis, Design Entry, or Pin filters in the Node Finder. The top-level ports of megafunctions are found using these three filters.

In contrast, timing simulation in the Quartus II software verifies the operation of your design with annotated timing information. This simulation is performed using the post place-and-route netlist. When performing a timing simulation, add only signals that exist after place and route. These signals are found with the Post-Compilation filter of the Node Finder. During synthesis and place-and-route, the names of your RTL signals change. Therefore, it might be difficult to find signals from your megafunction instantiation in the Post-Compilation filter. However, if you preserve the names of your signals during the synthesis and place-and-route stages, you must use the synthesis attributes `keep` or `preserve`. These are Verilog and VHDL synthesis attributes that direct analysis and synthesis to keep a particular wire, register, or node intact. Use these synthesis attributes to keep a combinational logic node so you can observe the node during simulation. More information about these attributes is available in volume 1 of the *Quartus II Handbook*.

EDA Simulation

Depending on the simulation tool you are using, refer to the appropriate chapter in volume 3 of the *Quartus II Handbook*. The *Quartus II Handbook* chapters show you how to perform functional and gate-level timing simulations that include the megafunctions, with details on the files that are needed and the directories where those files are located.

SignalTap II Embedded Logic Analyzer

The SignalTap® II embedded logic analyzer provides you with a non-intrusive method of debugging all of the Altera megafunctions within your design. With the SignalTap II embedded logic analyzer, you can capture and analyze data samples for the top-level ports of the Altera megafunctions in your design while your system is running at full speed.

To monitor signals from your Altera megafunctions, you must first configure the SignalTap II embedded logic analyzer in the Quartus II software, and then include the analyzer as part of your Quartus II project. The Quartus II software then embeds the analyzer with your design in the selected device seamlessly.

For more information about using the SignalTap II embedded logic analyzer, refer to volume 3 in the *Quartus II Handbook*.

Design Example: 4-Bit Absolute Function

This section presents a design example that uses the lpm_abs megafunction to generate an absolute value function. This example uses the MegaWizard Plug-In Manager in the Quartus II software. As you go through the wizard, each page is described in detail. When you are finished with this example, you can incorporate it into an overall design.

Design Files

The example design files are available in the Quartus II Projects section on the Design Examples page of the Altera web site www.altera.com.

Example

The objective of this design is to calculate the absolute value of a 4-bit signed input.

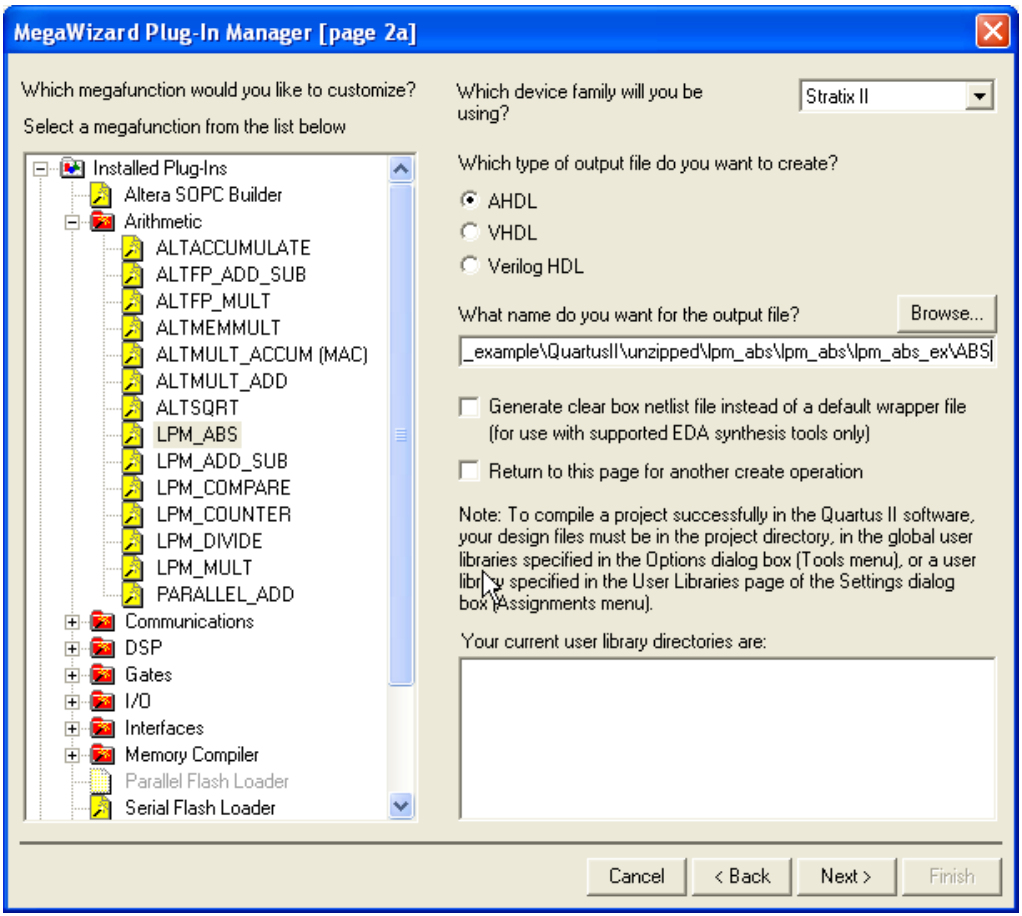
In this example, you perform the following activities:

- Generate a 4-bit absolute function using the lpm_abs megafunction
- Implement the absolute value function by assigning the EP2S30F484 device to the project and compiling the project
- Simulate the 4-bit absolute function design

Generate a 4-Bit Absolute Function

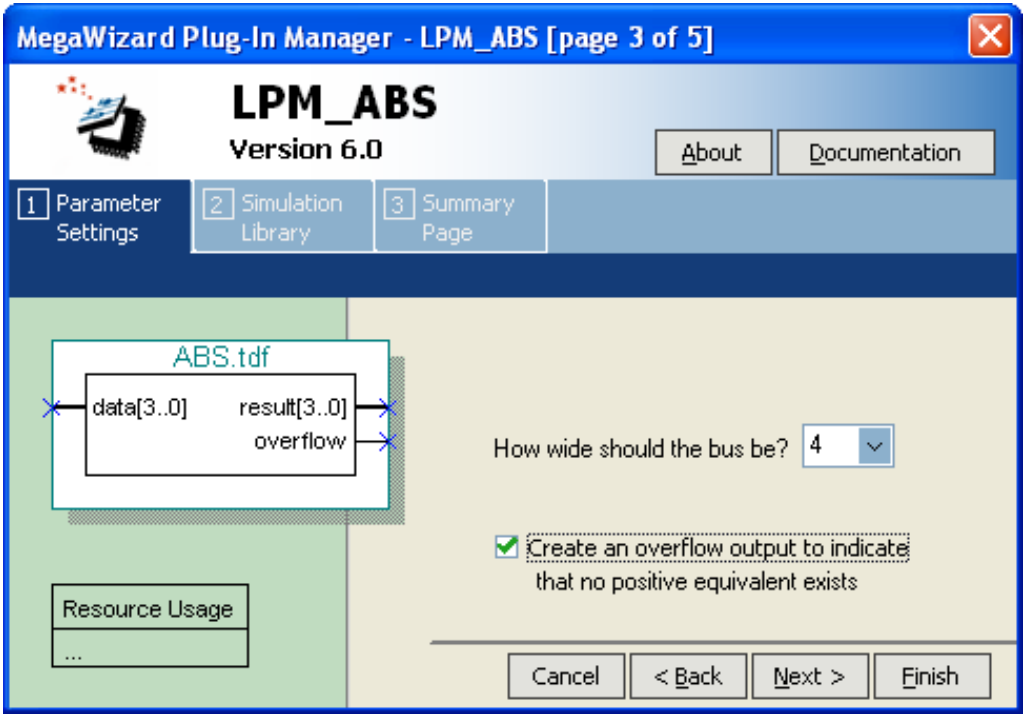
1. Open the project file **\lpm_abs_ex\lpm_abs_ex.qpf**.
2. Open the top-level file **\lpm_abs_ex\lpm_abs_ex.bdf**. This is an incomplete file that you will complete as part of this example.
3. Double-click on a blank area in the block design file (**.bdf**).
4. Click **MegaWizard Plug-In Manager** in the Symbol window, or on the Tools menu, choose **MegaWizard Plug-In Manager**. Page 1 of the MegaWizard appears.
5. On Page 1, in the **Which action do you want to perform?** section, select **Create a new custom megafunction variation**.
6. Click **Next**. Page 2a appears.
7. On Page 2a of the MegaWizard Plug-In Manager, expand the Arithmetic folder and select `LPM_ABS` (Figure 2–3).

Figure 2–3. MegaWizard Plug-In Manager [Page 2a]


MegaWizard Plug-In Manager [page 2a]
Which megafunction would you like to customize?
Select a megafunction from the list below
Installed Plug-Ins
Altera SOPC Builder
Arithmetic
ALTACCUMULATE
ALTFP_ADD_SUB
ALTFP_MULT
ALTMEMMULT
ALTMULT_ACCUM (MAC)
ALTMULT_ADD
ALTSQRT
LPM_ABS
LPM_ADD_SUB
LPM_COMPARE
LPM_COUNTER
LPM_DIVIDE
LPM_MULT
PARALLEL_ADD
Communications
DSP
Gates
I/O
Interfaces
Memory Compiler
Parallel Flash Loader
Serial Flash Loader
Which device family will you be using?
Stratix II
Which type of output file do you want to create?
AHDL
VHDL
Verilog HDL
What name do you want for the output file?
Browse...
_example\QuartusII\unzipped\lpm_abs\lpm_abs\lpm_abs_ex\ABS
Generate clear box netlist file instead of a default wrapper file (for use with supported EDA synthesis tools only)
Return to this page for another create operation
Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).
Your current user library directories are:
Cancel
< Back
Next >
Finish

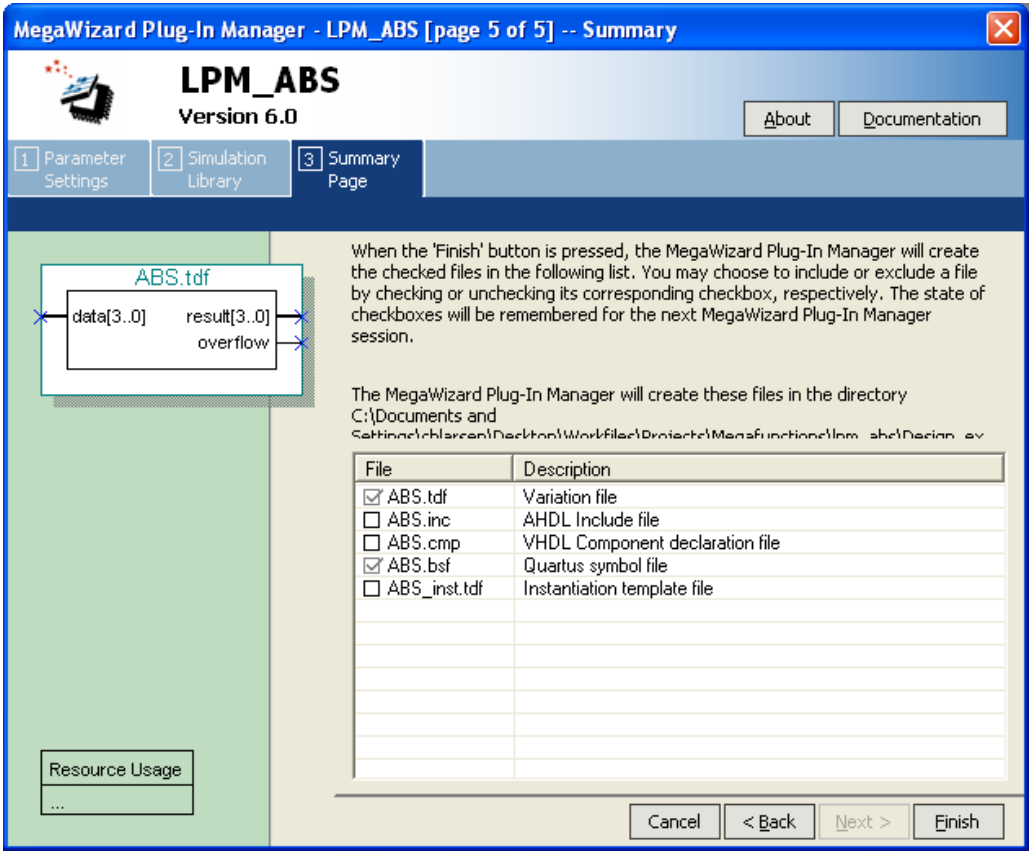
8. In the **Which device family will you be using?** list, select **Stratix II**.
9. In the **Which type of output file do you want to create?** section, select the **AHDL** option.
10. Name the output file **\lpm_abs_ex\ABS**.
11. Click **Next**. Page 3 appears (Figure 2–4).

Figure 2–4. MegaWizard Plug-In Manager - LPM_ABS [Page 3 of 5]



12. On Page 3, for the **How wide should the bus be?** option, select **4**.
13. Select **Create an overflow output to indicate that no positive equivalent exists**.
14. Click **Finish**. Page 5 appears.

Figure 2–5. MegaWizard Plug-In Manager - LPM_ABS [Page 5 of 5]



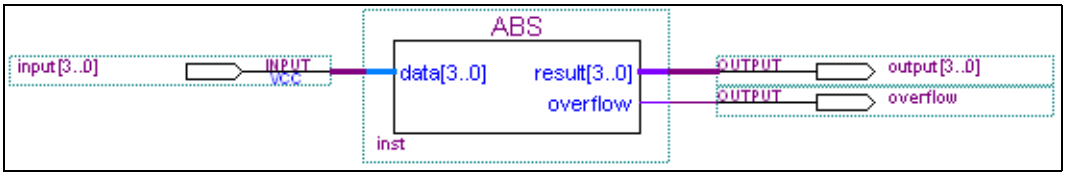
15. Make sure the Variation file (**.tdf**) and Quartus symbol file (**.bsf**) are selected and click **Finish**. The ABS module is built.

16. Move the pointer to place the ABS symbol in between the input/output ports in the **lpm_abs_ex.bdf** file. Click to place the symbol.

    You have now completed the design file as shown in Figure 2–6.

17. On the File menu, select **Save** to save the design.

Figure 2–6. lpm_abs Megafunction Design



Implement the 4-Bit Absolute Value Function

Next, assign the EP2S30F484C3 device to the project and compile the project.

1. To open the **Settings** dialog box, on the Assignments menu, select **Settings**.
2. Click the **Devices** category. In the Family field, ensure that Stratix II is selected.
3. In the **Target device** section, under the **Available devices** list, select **EP2S30F484C3**.
4. Click **OK**.
5. On the Processing menu, select **Start Compilation** or click the **Compilation** button to compile the design.
6. When the **Full Compilation was successful** message box appears, click **OK**.

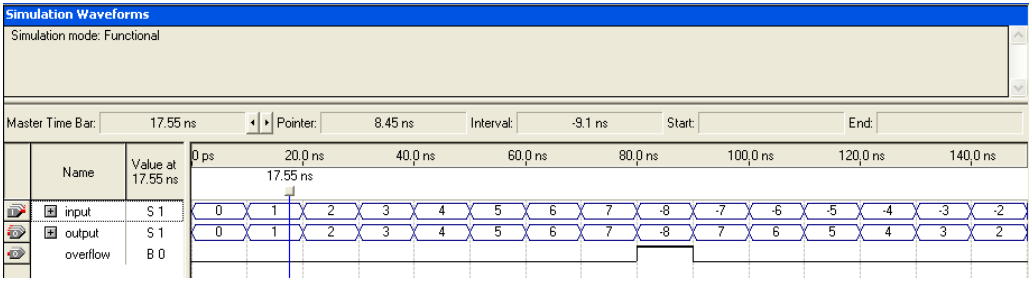
Functional Results—Simulate the 4-Bit Absolute Value in Quartus

Simulate the design to verify the results. Set up the Quartus II Simulator by performing the following steps.

1. On the Processing menu, select **Generate Functional Simulation Netlist.**
2. When the **Functional Simulation Netlist Generation was successful** message box appears, click **OK**.
3. To open the **Settings** dialog box, on the Assignments menu, select **Settings**.

4. In the **Category** list, select the **Simulator Settings** category.
5. Under **Simulation** mode, select **Functional**, and then select the required input vector waveform file (**lpm_abs_ex.vwf**). Make sure **Run simulation until all vector stimuli are used** is selected.
6. Click **Open** and then click **OK**.
7. On the Processing menu, select **Start Simulation**, or press **Ctrl+I**, or click on the **Simulation** button to run a simulation.
8. When the **Simulation was successful** message box appears, click **OK**.
9. In the **Simulation Report** window, view the simulation output waveforms and verify the results. Figure 2–7 shows the expected simulation results.

Figure 2–7. Simulation Waveforms



Functional Results—Simulate the 4-Bit Absolute Value in ModelSim-Altera

Simulate the design in ModelSim to compare the results of both simulators. Note that this ModelSim design example is for the ModelSim-Altera (Verilog) version.

This User Guide assumes that you are familiar with using ModelSim-Altera before trying out the design example. If you are unfamiliar, refer to http://www.altera.com/support/software/products/modelsim/mod-modelsim.html, which is a support page for ModelSim-Altera. There are various links to topics such as installation, usage, and troubleshooting.

Set up the ModelSim-Altera Simulator by performing the following steps.

1. Unzip the **lpm_abs_msim.zip** file to any working directory on your PC.

2. Browse to the folder in which you have unzipped the files and open the **lpm_abs_ex.do** file in a text editor.

3. In line 1, replace <*insert_directory_path_here*> with the directory path of the appropriate library files.

   For example,
   `C:/Modeltech_ae/altera/verilog/stratixii.`

4. On the File menu, select **Save**.

5. Start ModelSim-Altera.

6. On the File menu, select **Change Directory**.

7. Select the folder in which you have unzipped your files and click **OK**.

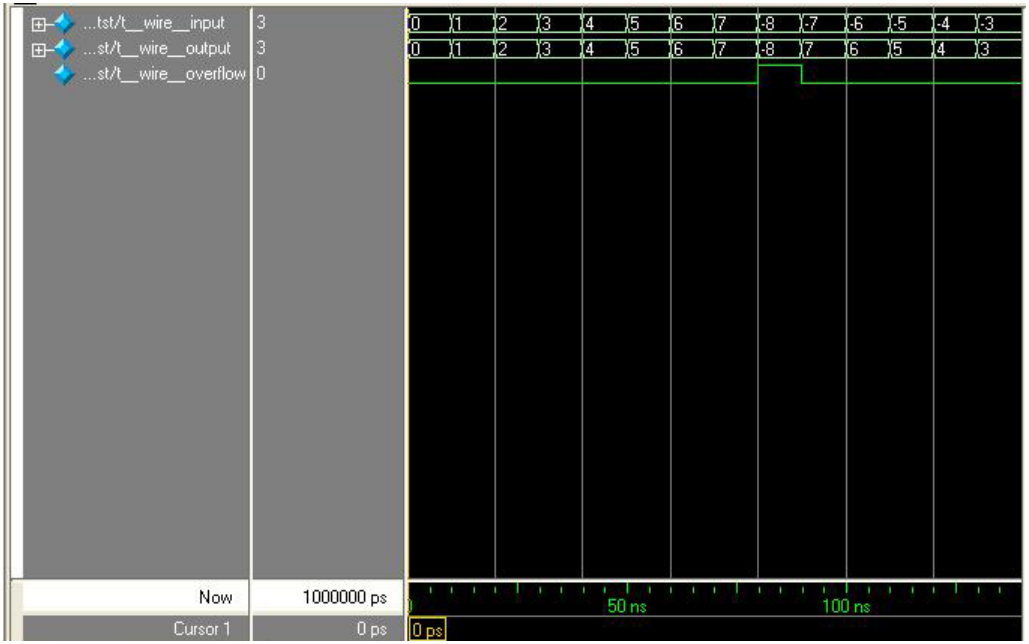
8. On the Tools menu, select **Execute Macro**.

9. Select the **lpm_abs_ex.do** file and click **Open**. This is a script file for ModelSim which automates all the necessary settings for the simulation.

10. Verify the results by looking at the **Waveform Viewer** window.

    You may need to rearrange signals, remove redundant signals, and change the radix to suit the results in the Quartus II Simulator. Figure 2–8 shows the expected simulation results in ModelSim.

Figure 2–8. ModelSim Simulation Results



Conclusion

The Quartus II software provides parameterizable megafunctions ranging from simple arithmetic units, such as adders and counters, to advanced phase-locked loop (PLL) blocks, multipliers, and memory structures. These megafunctions are performance-optimized for Altera devices and therefore, provide more efficient logic synthesis and device implementation, because they automate the coding process and save valuable design time. You should use these functions during design implementation so you can consistently meet your design goals.

Chapter 3. Specifications

t

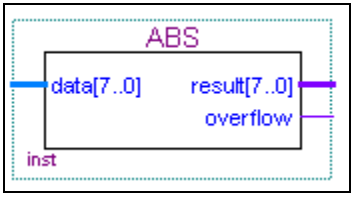
Ports and Parameters

Table 3–1 shows the input ports, Table 3–2 shows the output ports, and Table 3–3 shows lpm_abs megafunction parameters.

Refer to the latest version of the Quartus® II Help for the most current information on the ports and parameters for this megafunction.

The parameter details are only relevant for users who bypass the MegaWizard® Plug-In Manager interface and use the megafunction as a directly parameterized instantiation in their design. The details of these parameters are hidden from MegaWizard Plug-In Manager interface users.

Figure 3–1. lpm_abs Megafunction Port & Parameter Description Symbol



The lpm_abs megafunction ports and parameters tables follow.

Table 3–1. Input Ports Table

Port Name	Required	Description	Comments
data	Yes	Data input to the absoulte value function.	The width parameter specifies the size of the data[] port.

Table 3–2. Output Ports Table

Port Name	Required	Description	Comments
result[]	Yes	Data output from the absoulte value function.	The width parameter specifies the size of the result[] port.
overflow	No	Overflow output from the absolute value function.	

Table 3–3. Parameters Table

Parameter Name	Type	Required	Comments
WIDTH	Integer	Yes	Specifies the width of the data[] and the result[] ports.

.